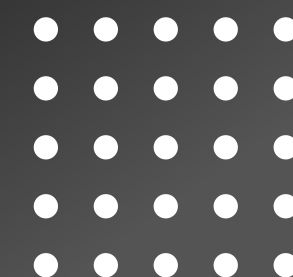


# TESTES DE SOFTWARE

Introdução





Prof. Thiago Suzuqui Lodi





# POR QUE TESTAR SOFTWARE?

- **Qualidade percebida pelo usuário**
    - software sem erros transmite confiança
  - **Redução de custos**
    - corrigir um bug em produção pode custar até 100x mais do que corrigir em desenvolvimento
  - **Prevenção de falhas críticas**
    - sistemas bancários, de saúde, e-commerces dependem de testes para não gerar prejuízos
- 
- 



# EXEMPLOS DE FALHAS REAIS

- **NASA** – Mars Climate Orbiter (1999)
  - perdeu uma sonda de \$125 milhões por causa de erro de unidade de medida (milhas X quilômetros)
- **Banco Itaú** (2012)
  - falha em atualização de sistema fez correntistas verem dinheiro a mais ou a menos
- **Facebook** (2021)
  - queda de 6h causou perda bilionária

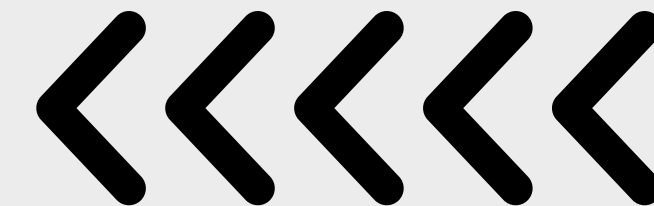




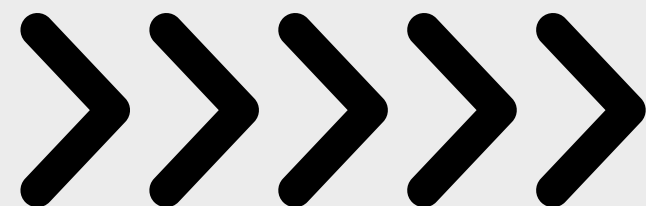
# QUALIDADE E CONFIANÇA DO USUÁRIO

- Usuários abandonam sistemas que falham constantemente
- Em app's notas ruins nas lojas geram menos downloads
- Software confiável gera vantagem competitiva
- Reflita: qual app/sistema você parou de usar porque "só dava problema"?

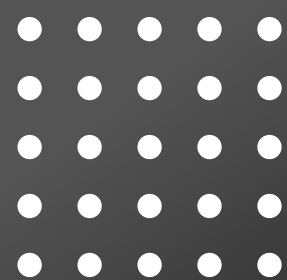
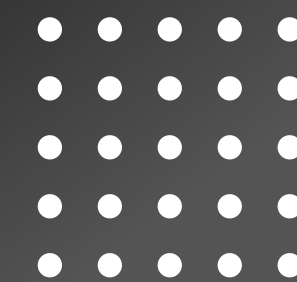




# FUNDAMENTOS





Prof. Thiago Suzuqui Lodi







# VERIFICAÇÃO E VALIDAÇÃO

- **Verificação** “Estamos construindo o software certo?”
    - Conferir se o sistema está conforme especificações e requisitos
  - **Validação** “Estamos construindo o software que o cliente realmente precisa?”
    - Confere se atende às necessidades reais do cliente/stakeholders
  - Um site pode estar corretamente implementado (verificação), mas se o cliente queria login com Google e só tem login por e-mail, falhou na validação.
- 
- 



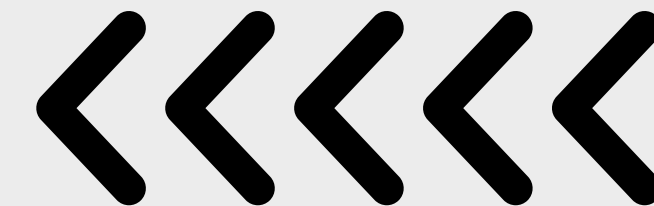
# QUALIDADE X CUSTO DE FALHAS

- Ciclo de vida de custo de bugs
    - quanto mais tarde se descobre um bug, mais caro fica sua correção.
  - Falha em requisitos
    - corrigida com ajuste na documentação
  - Falha em produção
    - pode gerar retrabalho na maioria das vezes
    - multas, indenizações
    - perda de clientes/contratos
- 
- 

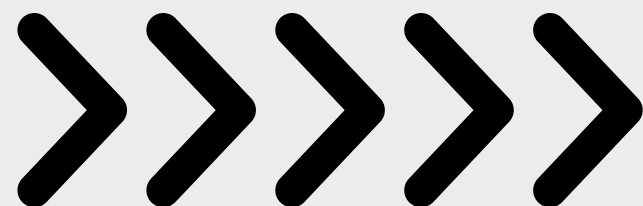
# CONCEITO DE BUG

- **Erro** – ação equivocada do programador
  - digitar “=” em vez de “==”
  - esquecer de validar campo obrigatório
- **Defeito** – problema no código
  - cálculo errado
  - regra de negócio com incoerências
  - usuário consegue salvar cadastro vazio
- **Bug** – quando o sistema apresenta comportamento incorreto em execução
- Ou seja, **erro** → **defeito** → **bug**





# TIPOS DE TESTES



Prof. Thiago Suzuqui Lodi







# FUNCIONAIS E NÃO FUNCIONAIS

- **Funcionais**

- verificam se o software faz o que deveria.
- Ex: login aceita usuário válido, bloqueia inválido.

- **Não funcionais**

- verificam como o software se comporta.
  - Ex: tempo de resposta, segurança, usabilidade.
- 
- 



# ESCOPO

- **Unitário**

- Menor parte testável do código
  - funções e métodos
- Dev

- **Integração**

- Verificam se os módulos funcionam como esperado juntos
- Dev

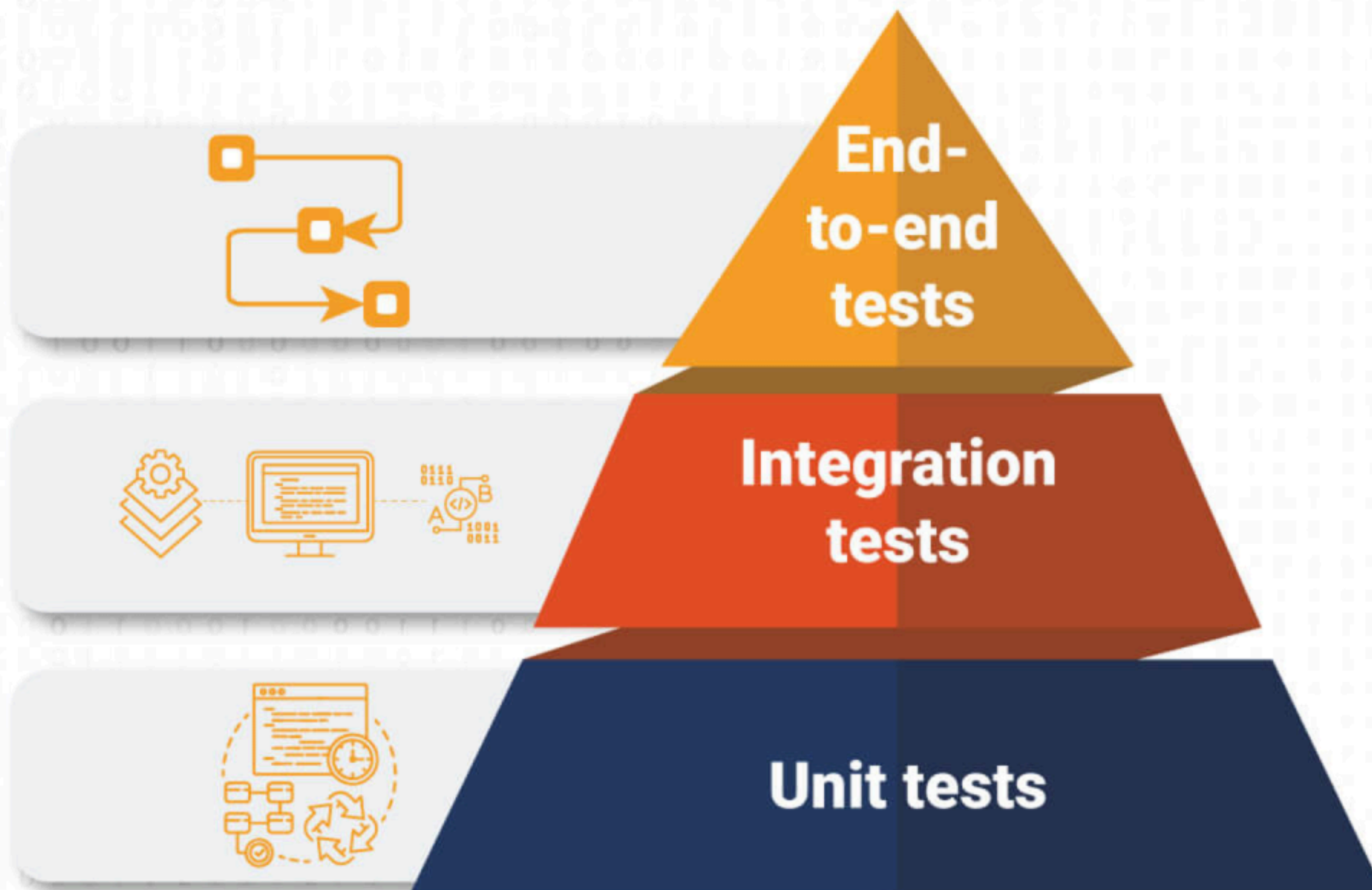
- **Sistema**

- Testam a aplicação completa de ponta a ponta
- Time de desenvolvimento

- **Aceitação**

- Deve validar os requisitos
- Cliente ou usuário final





# ATIVIDADE

Dada uma simples função que recebe 3 parâmetros inteiros e retorna a média dos valores para compor a nota semestral dos alunos, a assinatura da função é: `calcularMedia(n1, n2, n3);`  
Crie um planejamento de testes contemplando os cenários possíveis, ex: combinação de entrada a, b, c resulta em uma saída X.

# ATIVIDADE

Dada uma simples função que recebe 3 parâmetros inteiros e retorna a média dos valores para compor a nota semestral dos alunos, a assinatura da função é: `calcularMedia(n1, n2, n3)`; Crie um planejamento de testes contemplando os cenários possíveis, ex: combinação de entrada a, b, c resulta em uma saída X.

1. Média de 3 valores normais (ex: 7, 8, 9 → resultado 8).
2. Notas iguais (ex: 10, 10, 10 → resultado 10).
3. Notas mínimas (0, 0, 0 → resultado 0).
4. Valores fora do esperado (ex: nota negativa, nota >10).